数据可视化技术实验
实验一（2023.12.5）
大数据 2101 班　李嘉鹏　U202115652

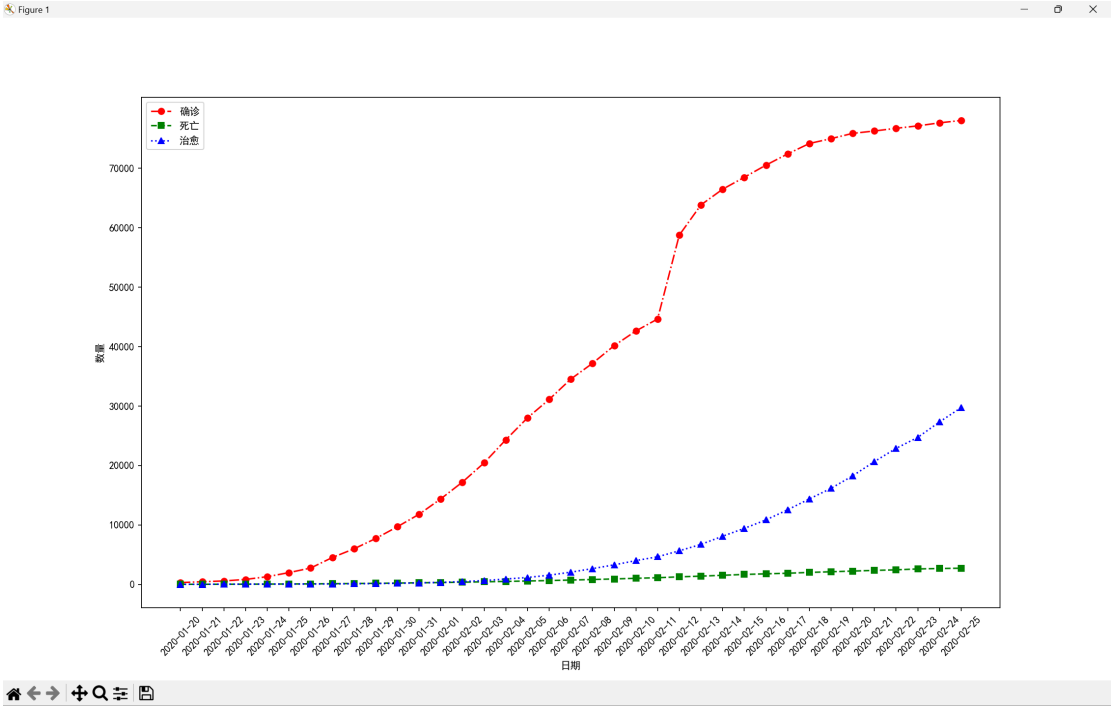
1. 利用折线图和散点图绘制 excel 文件中 data_history 对应的数据（按日期的新冠疫情数据），要求分别在折线图和散点图上显示 confirm，dead 和 heal 数据，使用不同的视觉通道（样式、颜色等）。
注意：（1）中文标注的使用；
　　（2）xticks 和 yticks 对坐标轴的处理。
思考：哪一个图更为有效？**（折线图，因为可以反映数据的变化趋势）**

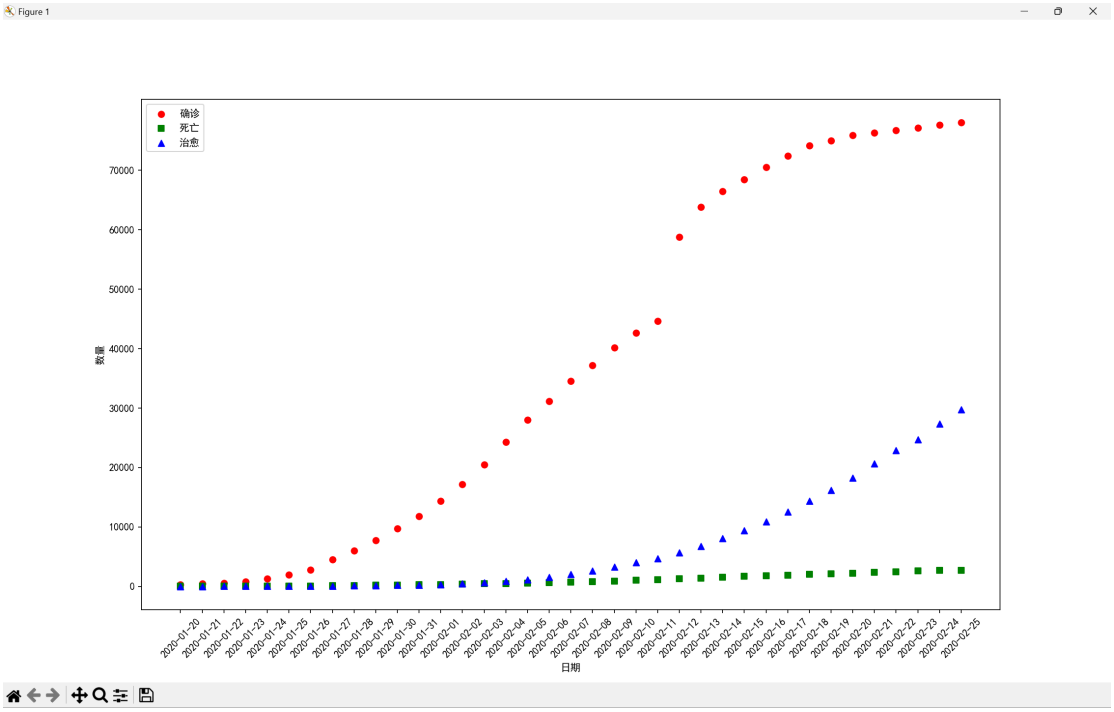
代码：

```
import pandas as pd
import matplotlib.pyplot as plt

# 读取 excel 文件
file_path = 'covid19_data.xls'
data = pd.read_excel(file_path, sheet_name='data_history')

# 设置中文显示
plt.rcParams['font.sans-serif'] = ['SimHei']

# 绘制折线图
plt.figure(figsize=(10, 5))
plt.plot(data['date'], data['confirm'], label='确诊', color='r',
linestyle='-.', marker='o')
plt.plot(data['date'], data['dead'], label='死亡', color='g',
linestyle='--', marker='s')
plt.plot(data['date'], data['heal'], label='治愈', color='b',
linestyle=':', marker='^')

# 设置坐标轴标签
plt.xlabel('日期')
plt.ylabel('数量')

# 设置坐标轴刻度
plt.xticks(rotation=45)
plt.yticks(range(0, max(data['confirm'])+1000, 10000))

# 显示图例
plt.legend()

# 显示折线图
plt.show()
```

```
# 绘制散点图
plt.figure(figsize=(10, 5))
plt.scatter(data['date'], data['confirm'], label='确诊', color='r',
marker='o')
plt.scatter(data['date'], data['dead'], label='死亡', color='g',
marker='s')
plt.scatter(data['date'], data['heal'], label='治愈', color='b',
marker='^')

# 设置坐标轴标签
plt.xlabel('日期')
plt.ylabel('数量')

# 设置坐标轴刻度
plt.xticks(rotation=45)
plt.yticks(range(0, max(data['confirm'])+1000, 10000))

# 显示图例
plt.legend()

# 显示散点图
plt.show()
```

结果:





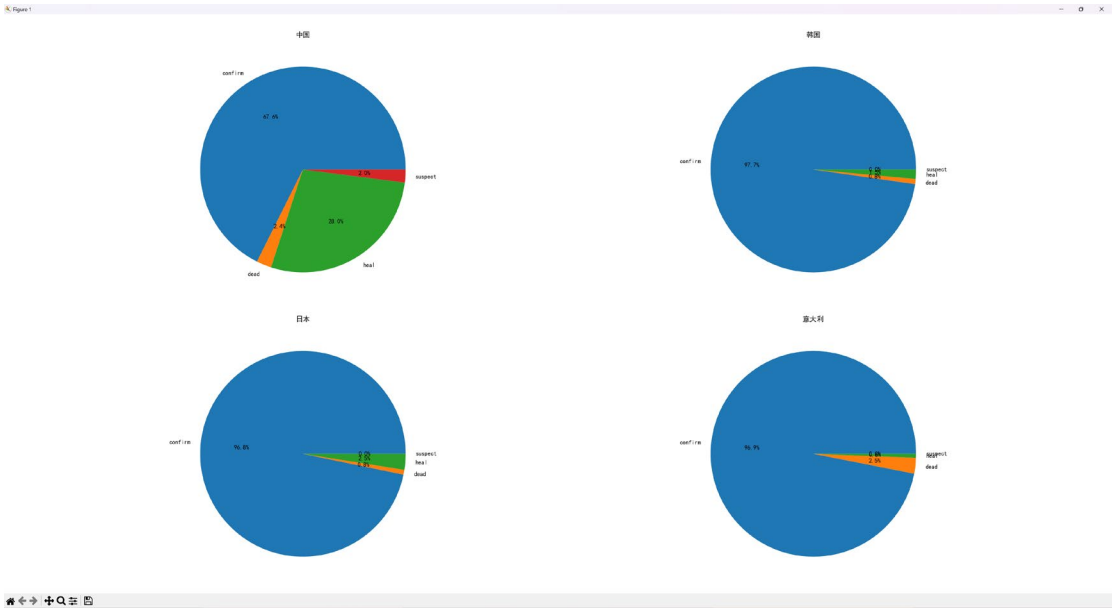
2. 利用饼图绘制 excel 文件中 data_world 对应的数据（各国新冠疫情数据），要求显示确诊人数最多的前 4 个国家的 confirm，dead、heal 和 suspect 的分布饼图。

代码：

```
import pandas as pd
import matplotlib.pyplot as plt

# 读取 Excel 文件
data = pd.read_excel('covid19_data.xls', sheet_name='data_world')

# 按确诊人数降序排序，取前 4 个国家
top_countries = data.nlargest(4, 'confirm')

# 设置中文显示
plt.rcParams['font.sans-serif'] = ['SimHei']

# 创建一个包含 4 个子图的画布
fig, axes = plt.subplots(nrows=2, ncols=2, figsize=(12, 8))

# 绘制每个国家的饼图
for i, country in enumerate(top_countries['country']):
    # 获取该国家的 confirm、dead、heal 和 suspect 数据
    confirm = top_countries.loc[top_countries['country'] == country,
'confirm'].values[0]
    dead = top_countries.loc[top_countries['country'] == country,
'dead'].values[0]
    heal = top_countries.loc[top_countries['country'] == country,
```

```
'heal'].values[0]
    suspect = top_countries.loc[top_countries['country'] == country,
'suspect'].values[0]

    # 绘制饼图
    ax = axes[i // 2, i % 2]
    ax.pie([confirm, dead, heal, suspect], labels=['confirm', 'dead',
'heal', 'suspect'], autopct='%1.1f%%')
    ax.set_title(country)

# 调整子图之间的间距
plt.tight_layout()

# 显示图形
plt.show()
```

结果:



3. 利用直方图和条形图绘制 excel 文件中 current_prov 对应的数据（各省新冠疫情数据），要求使用多个子图，使用合适的视觉通道。
思考：哪一个图更为有效？**（条形图，因为可以比较两组或以上的数值，找出其联系。但是在此题里实际上直方图和条形图的效果都比较差，因为横坐标“省份”相互独立，彼此之间没有关系；并且由于“湖北”的相关数据比其它省份的数据大很多，因此后面一些省份的数据都看起来很少）**

代码:

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
```

```
# 读取 excel 文件
file_path = 'covid19_data.xls'
current_prov = pd.read_excel(file_path, sheet_name='current_prov')

# 获取需要可视化的数据，例如各省的确诊人数、治愈人数和疑似人数
provinces = current_prov['province']
confirm_data = current_prov['confirm']
heal_data = current_prov['heal']
dead_data = current_prov['dead']

plt.rcParams['font.sans-serif'] = ['SimHei']

# 创建画布和子图
fig, axs = plt.subplots(4, 1, figsize=(10, 10))

# 条形图
index = np.arange(len(provinces))  # 省份的索引
bar_width = 0.2  # 条形宽度

# 条形图：各省确诊人数
rects1 = axs[0].bar(index - bar_width, confirm_data, bar_width,
color='skyblue', label='确诊人数')

# 条形图：各省治愈人数
rects2 = axs[0].bar(index, heal_data, bar_width, color='lightcoral',
label='治愈人数')

# 条形图：各省死亡人数
rects3 = axs[0].bar(index + bar_width, dead_data, bar_width,
color='green', label='死亡人数')

axs[0].set_xlabel('省份')
axs[0].set_ylabel('人数')
axs[0].set_title('各省确诊、治愈和死亡人数对比')
axs[0].set_xticks(index)
axs[0].set_xticklabels(provinces)
axs[0].legend()

# 直方图：各省确诊人数
axs[1].bar(provinces, confirm_data, color='skyblue')
axs[1].set_xlabel('省份')
axs[1].set_ylabel('确诊人数')
axs[1].set_title('各省确诊人数')
```

```
axs[2].bar(provinces, heal_data, color='lightcoral')
axs[2].set_xlabel('省份')
axs[2].set_ylabel('治愈人数')
axs[2].set_title('各省治愈人数')

axs[3].bar(provinces, dead_data, color='green')
axs[3].set_xlabel('省份')
axs[3].set_ylabel('死亡人数')
axs[3].set_title('各省死亡人数')

# 调整布局
plt.tight_layout()

# 显示图形
plt.show()
```

结果: